

: 1

```

LL               IIIIII               SSSSSSSS
LL               IIIIII               SSSSSSSS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SSSSSS
LL               II                   SSSSSS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SSSSSS
LLLLLLLLLLLLLL IIIIIIII               SSSSSSSS
LLLLLLLLLLLLLL IIIIIIII               SSSSSSSS

```

```
0001 0 XTITLE 'Julian Half Day Conversions'
0002 0 MODULE EVLJULIAN (
0003 0     LANGUAGE (BLISS32),
0004 0     IDENT = 'V04-000'
0005 0 ) =
0006 1 BEGIN
0007 1
0008 1
0009 1 *****
0010 1 *
0011 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0012 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0013 1 *  ALL RIGHTS RESERVED.
0014 1 *
0015 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0016 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0017 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0018 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0019 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0020 1 *  TRANSFERRED.
0021 1 *
0022 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0023 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0024 1 *  CORPORATION.
0025 1 *
0026 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0027 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0028 1 *
0029 1 *
0030 1 *****
0031 1
0032 1
0033 1 ++
0034 1 FACILITY:    DECnet Event Logging (EVL)
0035 1
0036 1 ABSTRACT:
0037 1
0038 1     This module contains the routines to convert to and from
0039 1     the standard date-time format for event logging, Julian
0040 1     halfday. The internal date-time for DECnet-VAX is VAX
0041 1     64 bit absolute time.
0042 1
0043 1 ENVIRONMENT: VAX/VMS Operating System
0044 1
0045 1 AUTHOR:      Darrell Duffy , CREATION DATE: 8-Jun-1980
0046 1
0047 1 MODIFIED BY:
0048 1
0049 1
0050 1 01 : VERSION
0051 1 --
```

```
.. 53 0052 1 %SBTTL 'Definitions'
.. 54 0053 1
.. 55 0054 1
.. 56 0055 1 : TABLE OF CONTENTS:
.. 57 0056 1
.. 58 0057 1
.. 59 0058 1 FORWARD ROUTINE
.. 60 0059 1 EVL$JULIAN : NOVALUE : Convert from abstim to julian
.. 61 0060 1 EVL$UNJULIAN : NOVALUE : Convert from julian to abstim
.. 62 0061 1 ;
.. 63 0062 1
.. 64 0063 1
.. 65 0064 1 : INCLUDE FILES:
.. 66 0065 1
.. 67 0066 1
.. 68 0067 1 LIBRARY 'SYSS$LIBRARY:STARLET.L32';
.. 69 0068 1
.. 70 0069 1
.. 71 0070 1 : MACROS:
.. 72 0071 1
.. 73 0072 1
.. 74 0073 1
.. 75 0074 1 : EQUATED SYMBOLS:
.. 76 0075 1
.. 77 0076 1
.. 78 0077 1 LITERAL
.. 79 0078 1 SUCCESS = 1,
.. 80 0079 1 FAILURE = 0
.. 81 0080 1 ;
.. 82 0081 1
.. 83 0082 1
.. 84 0083 1 : OWN STORAGE:
.. 85 0084 1
.. 86 0085 1
.. 87 0086 1
.. 88 0087 1 : EXTERNAL REFERENCES:
.. 89 0088 1
.. 90 0089 1
.. 91 0090 1 : EXTERNAL ROUTINE
.. 92 0091 1 ;
```

```

94 0092 1 XSBTTL 'EVL$JULIAN Convert Abstim to Julian Half Days'
95 0093 1 GLOBAL ROUTINE EVL$JULIAN (ABSTIM, HALFDAY, SECONDS, MILLISEC) =
96 0094 1
97 0095 1 ++
98 0096 1 FUNCTIONAL DESCRIPTION:
99 0097 1
100 0098 1 Convert from VMS abs time to julian half day, seconds and
101 0099 1 milliseconds. This computation is taken directly from the
102 0100 1 DNA Network Management Functional Specification.
103 0101 1
104 0102 1 FORMAL PARAMETERS:
105 0103 1
106 0104 1 ABSTIM Address of quadword abs time
107 0105 1 HALFDAY Address to return halfday as a longword
108 0106 1 SECONDS Address to return seconds in half day as a longword
109 0107 1 MILLISEC Address to return milliseconds as a longword
110 0108 1
111 0109 1 IMPLICIT INPUTS:
112 0110 1
113 0111 1 NONE
114 0112 1
115 0113 1 IMPLICIT OUTPUTS:
116 0114 1
117 0115 1 NONE
118 0116 1
119 0117 1 ROUTINE VALUE:
120 0118 1 COMPLETION CODES:
121 0119 1
122 0120 1 Success if data returned, Failure if abs time is out of range
123 0121 1 of julian half day, or conversion of abstime fails.
124 0122 1
125 0123 1 SIDE EFFECTS:
126 0124 1
127 0125 1 NONE
128 0126 1
129 0127 1 --
130 0128 1
131 0129 1 BEGIN
132 0130 2
133 0131 2 LOCAL
134 0132 2 TIMVEC : VECTOR [7, WORD], ! Vector of words to return dissected
135 0133 2 STATUS ! Abs time
136 0134 2 ! Local status
137 0135 2 ;
138 0136 2
139 0137 2 BIND
140 0138 2 YEAR = TIMVEC [0] : WORD, ! Each piece of the dissected time
141 0139 2 MONTH = TIMVEC [1] : WORD,
142 0140 2 DAY = TIMVEC [2] : WORD,
143 0141 2 HOUR = TIMVEC [3] : WORD,
144 0142 2 MINUTE = TIMVEC [4] : WORD,
145 0143 2 SECND = TIMVEC [5] : WORD,
146 0144 2 HNDRTH = TIMVEC [6] : WORD
147 0145 2 ;
148 0146 2
149 0147 2 IF NOT ! Disect the abs time
150 P 0148 3 (STATUS = $NUMTIM
```

```
151 P 0149 (
152 P 0150 TIMBUF = TIMVEC, ! Buffer to place dissected time
153 P 0151 TIMADR = .ABSTIM ! Place to obtain 64 bit time
154 0152 )
155 0153 )
156 0154 THEN
157 0155 RETURN .STATUS ! It was not valid
158 0156 ;
159 0157
160 0158 IF ( ! Check the range of the date
161 0159 .YEAR GTRU 2021
162 0160 AND
163 0161 .MONTH GTR 10
164 0162 )
165 0163 OR
166 0164 .YEAR LSSU 1977
167 0165 THEN
168 0166 RETURN FAILURE ! Not expressible in julian halfday
169 0167 ;
170 0168
171 0169 .HALFDAY = ! Compute the half day
172 0170 (
173 0171 ( (3055 * (.MONTH+2) / 100) - ( (.MONTH+10) / 13) * 2 - 91)
174 0172 +
175 0173 ( (1 - (.YEAR - .YEAR / 4 * 4 + 3) / 4) * (.MONTH+10) / 13 + .DAY - 1)
176 0174 +
177 0175 ( (.YEAR-1977) * 365 + (.YEAR-1977) / 4)
178 0176 ) * 2
179 0177 ;
180 0178
181 0179 .HALFDAY = .HALFDAY + (.HOUR/12); ! Adjust for the odd half day
182 0180 HOUR = .HOUR MOD 12;
183 0181
184 0182 .SECONDS = ( .HOUR*3600 + .MINUTE*60 + .SECND ); ! Now the second in day
185 0183
186 0184 .MILISEC = .HNDRTH * 10; ! And the millisecond in the second
187 0185
188 0186 RETURN SUCCESS
189 0187
190 0188 END;
```

				.TITLE	EVLJULIAN Julian Half Day Conversions	
				.IDENT	\V04-000\	
				.EXTRN	SYSSNUMTIM	
				.PSECT	\$CODE\$,NOWRT,2	
				.ENTRY	EVL\$JULIAN, Save R2,R3,R4	: 0093
				SUBL2	#16, SP	
				PUSHL	ABSTIM	: 0152
				PUSHAB	TIMVEC	
				CALLS	#2, SYSSNUMTIM	
				BLBS	STATUS, 1\$	
				RET		: 0159
				MOVZWL	YEAR, R4	
00000000G	00	04	10	001C	00000	
	01	04	AC	DD	00002	
			AE	9F	00005	
			02	FB	00008	
			50	EB	00012	
				04	00015	
	54		6E	3C	00016	1\$:

07E5	8F	54	B1	00019	CMPW	R4, #2021	
		09	1B	0001E	BLEQU	3\$	
	0A	02	AE	B1	CMPW	MONTH, #10	0161
		03	1B	00024	BI.EQU	3\$	
		00BB	31	00026	BRW	4\$	
07B9	8F	54	B1	00029	CMPW	R4, #1977	0164
		F6	1F	0002E	BLSSU	2\$	
	52	02	AE	3C	MOVZWL	MONTH, R2	0171
	52	00000BEF	8F	C4	MULL2	#3055, R2	
	52	17DE	C2	9E	MOVAB	6110(R2), R2	
	52	00000064	8F	C6	DIVL2	#100, R2	
	53	02	AE	3C	MOVZWL	MONTH, R3	
	53		0A	C0	ADDL2	#10, R3	
50	53		0D	C7	DIVL3	#13, R3, R0	
	50		02	C4	MULL2	#2, R0	
	52		50	C2	SUBL2	R0, R2	
51	54		04	C7	DIVL3	#4, R4, R1	0173
	51		04	C4	MULL2	#4, R1	
	51		54	C2	SUBL2	R4, R1	
	51		03	C2	SUBL2	#3, R1	
	51		04	C6	DIVL2	#4, R1	
	50	01	A1	9E	MOVAB	1(R1), R0	
	50		53	C4	MULL2	R3, R0	
	50		0D	C6	DIVL2	#13, R0	
	51	04	AE	3C	MOVZWL	DAY, R1	
	50		51	C0	ADDL2	R1, R0	
	50		52	C0	ADDL2	R2, R0	0172
51	54	0000016D	8F	C5	MULL3	#365, R4, R1	0175
	52	F847	C4	9E	MOVAB	-1977(R4), R2	
	52		04	C6	DIVL2	#4, R2	
	51		52	C0	ADDL2	R2, R1	
	50	FFF4FCDF	E140	9E	MOVAB	-721697(R1)[R0], R0	0170
08	BC		01	78	ASHL	#1, R0, @HALFDAY	0176
			AE	3C	MOVZWL	HOUR, R0	0179
		06	0C	C6	DIVL2	#12, R0	
	08	BC	50	C0	ADDL2	R0, @HALFDAY	
			AE	3C	MOVZWL	HOUR, R0	0180
7E	00		01	7A	EMUL	#1, R0, #0, -(SP)	
50	50		0C	7B	EDIV	#12, (SP)+, R0, R0	
	06		50	B0	MOVW	R0, HOUR	
			AE	3C	MOVZWL	HOUR, R0	0182
	50	06	8F	C4	MULL2	#3600, R0	
	50	00000E10	AE	3C	MOVZWL	MINUTE, R1	
	51	08	3C	C4	MULL2	#60, R1	
	51		51	C0	ADDL2	R1, R0	
	50		AE	3C	MOVZWL	SECND, R1	
0C	BC		51	C1	ADDL3	R1, R0, @SECONDS	0184
			AE	3C	MOVZWL	HNDRTH, R0	
10	BC		0A	C5	MULL3	#10, R0, @MILISEC	
			01	D0	MOVL	#1, R0	0186
			50	D4	RET		0188
			04	000E6	CLRL	R0	

; Routine Size: 231 bytes, Routine Base: \$CODE\$ + 0000

```
0189 1 ZSBTTL 'EVL$UNJULIAN Convert Julian Halfday to Abs Time'
0190 1 GLOBAL ROUTINE EVL$UNJULIAN (JULIAN, SECNDS, MILSECS, ABSTIM) :NOVALUE =
0191 1
0192 1 ++
0193 1 FUNCTIONAL DESCRIPTION:
0194 1
0195 1 Convert julian halfday, seconds and milliseconds to VMS 64 bit
0196 1 absolute time. We need to do lots of monkeying around to not have
0197 1 the one EMUL instruction overflow. The important conversion factor
0198 1 in this computation is the number of days between 17-NOV-1858 and
0199 1 1-JAN-1977.
0200 1
0201 1 FORMAL PARAMETERS:
0202 1
0203 1 JULIAN Address of longword containing julian halfdays
0204 1 SECNDS Address of longword containing seconds in halfday
0205 1 MILSECS Address of longword containing milliseconds in second
0206 1 ABSTIM Address of quadword for abs time
0207 1
0208 1 IMPLICIT INPUTS:
0209 1
0210 1 NONE
0211 1
0212 1 IMPLICIT OUTPUTS:
0213 1
0214 1 NONE
0215 1
0216 1 ROUTINE VALUE:
0217 1 COMPLETION CODES:
0218 1
0219 1 NONE
0220 1
0221 1 SIDE EFFECTS:
0222 1
0223 1 NONE
0224 1
0225 1 --
0226 1
0227 1 BEGIN
0228 1
0229 1 BUILTIN EMUL ; ! Extended multiply instruction
0230 1
0231 1 LOCAL
0232 1 NANOSECS, ! 100 nanosecs to add
0233 1 JULIAN_MINS, ! Minutes since 1-jan-1977
0234 1 NANOSPERMIN ! 100 nanosecs in a minute
0235 1 ;
0236 1
0237 1 BIND
0238 1 DATEOFFSET = 43144 ! Days between 17-NOV-1858 and
0239 1 ; ! 1-Jan-1977
0240 1
0241 1 NANOSPERMIN = 60*10*1000*1000;
0242 1 NANOSECS = ( ( (..SECNDS MOD 60) *1000) + ..MILSECS ) * (10*1000);
0243 1 JULIAN_MINS = (..JULIAN + (DATEOFFSET*2) ) * (12*60) + (..SECNDS / 60);
0244 1 EMUL (JULIAN_MINS, NANOSPERMIN, NANOSECS, .ABSTIM)
0245 1
```

: 249 0246 1 END:

DATEOFFSET= 43144

[illegible]

.ENTRY	EVLSUNJULIAN, Save R2,R3	:	0190
MOVL	#600000000, NANOSPERMIN	:	0241
EMUL	#1, @SECNDS, #0, -(SP)	:	0242
EDIV	#60, (SP)+, R0, R0	:	
MULL2	#1000, R0	:	
ADDL2	@MILSECS, R0	:	
MULL3	#10000, R0, NANOSECS	:	
MULL3	#720, @JULIAN, R0	:	0243
DIVL3	#60, @SECNDS, R1	:	
MOVAB	62127360(R1)(R0), JULIAN_MINS	:	
EMUL	JULIAN_MINS, NANOSPERMIN, NANOSECS, @ABSTIM	:	0244
RET		:	0246

; Routine Size: 68 bytes, Routine Base: \$CODE\$ + 00E7

EVLJULIAN
V04-000

Julian Half Day Conversions
EVL\$UNJULIAN Convert Julian Halday to Abs Tim

L 6
16-Sep-1984 01:34:45
14-Sep-1984 12:28:48

VAX-11 Bliss-32 V4.0-742
[EVL.SRC]EVLJULIAN.B32;1

Page 8
(5)

: 251
: 252
0247 1 END
0248 0 ELUDOM
!End of module

PSECT SUMMARY

Name Bytes Attributes
\$CODE\$ 299 NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	3	0	581	00:01.0

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:EVLJULIAN/OBJ=OBJ\$:EVLJULIAN MSRC\$:EVLJULIAN/UPDATE=(ENH\$:EVLJULIAN)

: Size: 299 code + 0 data bytes
: Run Time: 00:05.6
: Elapsed Time: 00:13.2
: Lines/CPU Min: 2647
: Lexemes/CPU-Min: 9256
: Memory Used: 68 pages
: Compilation Complete

0156 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

